REMARKS

Please reconsider the application in view of the following remarks. Applicant thanks the Examiner for carefully considering this application.

Disposition of Claims

Claims 12, 13, 15-18, 20, 22, and 25-27 are pending. Claims 14, 21, 23, and 28 are canceled by this reply, without prejudice or disclaimer. Claims 12, 20, 22, and 27 are independent. The remaining claims depend, directly or indirectly, from claims 12, 20, 22, and 27.

Examiner Interview

Applicants thank the Examiner for courtesies extended during the Examiner Interview conducted on October 8, 2009. During the Examiner Interview, Applicants discussed proposed amendments to the claims and the Dwyer reference. No agreement was reached. In addition, the Examiner made suggestions for proactively addressing any potential 35 U.S.C. § 101 rejections of the claims.

Rejection(s) under 35 U.S.C. § 102

Claims 12-18 and 20 are rejected under 35 U.S.C. § 102(e) as being anticipated by US Pub. No. 2004/0015748 ("Dwyer"). Claim 14 is canceled; thus, this rejection is now moot with respect to the canceled claims. To the extent that this rejection may still apply to the amended claims, this rejection is respectfully traversed.

For anticipation under § 102, "[a] claim is anticipated only if *each and every element* as set forth in the claims is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631 (Fed. Cir. 1987) (emphasis added). Further, "[t]he identical invention must be shown in as complete detail as is contained in the claim." *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236 (Fed. Cir. 1989). Applicants assert that Dwyer fails to disclose each and every element of the claims for at least the following reasons.

The claimed invention relates to checking code portions of an executing program while the program is executing. According to the claimed invention, verification of a current code portion takes place first, and execution of the program is interrupted (does not continue to the next routine or next code portion) in the event that verification of the current code portion fails. Further, in the event that checksums are used to verify code portions, a pre-calculated checksum is computed during compilation of the program, which is then compared with a checksum that is calculated during execution of a code portion. If the checksums match, the program execution continues. If the checksums do not match, then program execution is interrupted. *See* publication of present application, paragraphs [0028] and [0034].

Accordingly, the amended independent claims require, in part, (i) detecting an anomaly when the checksum/counter is not equal to the pre-calculated checksum/counter values, wherein detection of the anomaly results in the second code portion remaining unexecuted. That is, the first code portion is executed first, and only the second code portion remains unexecuted if an anomaly exists. The first code portion still executes in the claimed invention, and is verified after execution.

Further, amended independent claims 12 and 20 require a first pre-calculated checksum is calculated during *compilation*.

In contrast to the claimed invention, Dwyer first checks the CRC of a block, and then only (if the check is successful), executes the block. *See* Dwyer, paragraph [0009]. That is, in Dwyer the CRC is checked first (in order to check whether the software is invalid) and only then can the execution proceed, if the software is found not to be invalid based on CRC check. Clearly, this is the opposite of the claimed invention, in which the first code portion is executed, then checked to ensure execution took place correctly, wherein the second code portion remains unexecuted if a problem is found with execution of the first code portion. In fact, Dwyer does not even execute the first code portion if the checksum verification fails.

It logically follows from the above that Dwyer does not interrupt program execution if verification of one block fails. While that current block may be skipped, program execution continues in Dwyer. Accordingly, Dwyer fails to disclose (i) as required above.

Further, the Examiner appears to have improperly equated execution of the compilation process for creating a compiled program, and execution of the compiled program, which are distinct processes. In Dwyer, the pre-computed checksums are verified during execution of the compiled program. However, during compilation in Dwyer, there is no verification of the checksums. Rather, in Dwyer, the checksums are computed, and some verification code is added in order to enable verification of the checksums at runtime. With respect to claims 12 and 20, the claimed invention compares two checksums – one that is pre-calculated during compilation, and one that is calculated during execution of the program. However, Dwyer fails to disclose a pre-calculated checksum that is calculated during compilation. The only disclosure of anything occurring during

compilation in Dwyer is the insertion of verification code, which is then executed when the compiled program executes. However, insertion of verification code is limited to adding code before compilation, and does not result in a pre-calculated checksum, as required by the claimed invention.

In view of the above, it is clear that the Examiner's contentions fail to support an anticipation rejection of the amended independent claims. Pending dependent claims are patentable for at least the same reasons. Accordingly, withdrawal of this rejection is respectfully requested.

Rejection(s) under 35 U.S.C. § 103

Claim 21 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Dwyer in view of US Pub. No. 2002/0023963 ("Luu"). Claim 21 is canceled; thus, this rejection is now moot.

Claims 22-23 and 25-27 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Dwyer in view of US Patent No. 5,586,321 ("Shavit"). Claim 23 is canceled; thus, this rejection is now moot with respect to claim 23. To the extent that this rejection may still apply to the amended claims, this rejection is respectfully traversed.

As described above, Dwyer fails to disclose or render obvious the limitations required by independent claims 12 and 20. Further, independent claims 22 and 27 include similar subject matter (*i.e.,* the execution of a code portion first, and subsequent verification of the executed code portion, where program execution is interrupted if an anomaly is detected). Accordingly, amended claims 22 and 27 are patentable over Dwyer for at least the same reasons described above with respect to claims 12 and 20.

In addition, Dwyer fails to disclose or render obvious using a counter to verify code. In fact, Dwyer does not even mention using a counter for any purpose. Applicant asserts that the Examiner, in relying on Dwyer as disclosing pre-calculated counter values and the incrementing of a counter by the value associated with each of the plurality of instructions executed during the execution of the first routine, has clearly read out specific limitations of the claimed invention. The Examiner's assertion is in violation of the guidelines set forth in *In re Wilson*, 424 F.2d 1382, 1385 (CCPA 1970) ("[a]ll words in a claim must be considered in judging the patentability of that claim against the prior art"). There are absolutely **no** counters used in Dwyer for verification of code; rather, Dwyer only discloses the use of checksums.

Further, Shavit fails to supply that which Dwyer lacks. That is, Shavit fails to disclose or render obvious verification of code after execution, and interruption of program execution when verification of one executed code portion fails. Shavit also fails to disclose using counters for verification of code. In fact, Shavit merely discloses a diffracting token router that routes tokens from one producer to two consumers. *See* Shavit, Abstract.

Moreover, Shavit fails to disclose or render obvious that which the Examiner relies on Shavit as teaching. The Examiner asserts that Shavit teaches balancing branch instructions in the first code segment such that counter values come out to be the same no matter which branch instruction is taken. However, Shavit is not even related to software code verification. Shavit is only focused on a diffracting token router, which is a hardware device that is involved with token collisions and routing of tokens to output wires. *See* Shavit, Abstract. Given Shavit's area of technology, it is impossible for Shavit to disclose or render obvious the balancing of instructions in branched code segments. The cited portion of Shavit merely discloses a shared counter that is used

as an application of the diffracting token router. The shared counter disclosed in Shavit is not, in any way, used to balance counter values when one or another branch is taken in first code segment. In fact, there is no mention of incrementing the shared counter of Shavit each time instructions in a branch sequence are executed.

In view of the above, it is clear that independent claims 22 and 27 are patentable over Dwyer and Shavit, whether considered separately or in combination. Pending dependent claims are patentable for at least the same reasons. Accordingly, withdrawal of this rejection is respectfully requested.

Claim 28 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Dwyer in view of Shavit and further in view of Luu. Claim 28 is canceled; thus, this rejection is now moot.

Conclusion

Applicant believes this reply is fully responsive to all outstanding issues and places this application in condition for allowance. If this belief is incorrect, or other issues arise, the Examiner is encouraged to contact the undersigned or his associates at the telephone number listed below. Please apply any charges not covered, or any credits, to Deposit Account 50-0591 (Reference Number 09669/087001).

Dated: October 22, 2009

Respectfully submitted,

By /Jonathan P. Osha/

Jonathan P. Osha
Registration No.: 33,986
OSHA · LIANG LLP
909 Fannin Street, Suite 3500
Houston, Texas 77010
(713) 228-8600
(713) 228-8778 (Fax)
Attorney for Applicant